

1/4

E1	E2	E3	E4
A	R	A	A

(7)

1. Podemos definir a  $f$  como

$$f(x, y) = \min_{S \in \text{MAX}} (|S| = |y|) \wedge (\forall i \leq |y|) [(|S[i]| = |x|) \wedge (\forall j \leq |x|) [S[j][i] = x[j] \cdot y[i]]]$$

Esto es, buscamos la mínima lista de listas ~~con~~ que cumpla con las dimensiones del resultado y cada posición contenga la multiplicación de la definición.

Hace falta definir la cota de la minimización, MAX. Para ello vemos que

$$\begin{cases} x[i] \cdot y[j] \leq x \cdot y & \forall i, j \\ |f(x, y)[i]| = |x| & \forall i \wedge |f(x, y)| = |y| \end{cases}$$

$$\Rightarrow f(x, y)[i] \leq x^{x \cdot y} \quad \forall i$$

$$\Rightarrow f(x, y) \leq y^{x^{x \cdot y}}$$

$$\Rightarrow \text{MAX} = y^{x^{x \cdot y}}$$

Finalmente, como pudimos definir  $f$  a componiendo funciones p.p.

(visto en la práctica),  $f$  es p.p.

PUTO EL QUE LEE

(R)

2/4

2. Verdadero. ✓

Sea  $x$  un número cualquiera. Definimos el programa  $P$  ~~como:~~ como:

$P:$   
 $Y \leftarrow \phi_x(X_1)$  ✓  
 $Y \leftarrow Y + 1$

Sea  $y = \#P$ .

Vemos claramente que

$$\phi_z(y) = \phi_x(y) + 1 \quad \forall y \quad (\text{si } \phi_x \text{ se cuelga, } \phi_z \text{ se cuelga}) \quad \checkmark$$

~~por lo tanto la guarda de  $f$  es verdadera~~

por lo tanto la guarda de  $f$  es verdadera para cualquier  $x$ , ✓  
 y  $f$  es la función constante 1 (computable por ser  $\text{pp}$ ).  $\square$  ✓

b. Verdadero ✓ (Mal justificado).

Escribimos uno de los infinitos  $P$  posibles a continuación. Llamaremos  $e$  al número del programa  $P$  (por el Teorema de la recursión, no hay problema con que  $P$  conozca su propio número).

$P:$  (X) No. Lo que afirma el teorema de la recursión es que se pueden encontrar programas que "conozcan" su propio número, en el sentido de que

[A] IF  $x \gg y$  GOTO A  
 $Y \leftarrow X_1 + X_2 + e$

consisten en evaluar una función parcial computable con su propio número. No se puede usar el número de un programa dentro de su propio código sin más: no es una instrucción válida de  $S$ , ni se puede definir como una macro (¿cómo lo harías?)

3/4

3. d.

Sea  $t$  tal que  $STP(y, x, t)$ . Llamemos  $P$  al programa de número  $x$ .

Como  $P$  terminó en  $t$  lo sumo  $t$  pasos, ~~esta~~ podemos definir a  $L$  como la lista de descripciones instantáneas de la ejecución de  $P$ , ~~esta~~ y tenemos

$$|L| \leq t$$

$$L = \{ \langle i_1, s_1 \rangle, \langle i_2, s_2 \rangle, \dots, \langle i_k, s_k \rangle \}, \quad k \leq t$$

donde  $i_j$  son los números de instrucción actual y  $s_j$  son los estados de las variables.

En  $s_1$  la variable  $Y$  vale 0

En  $s_{j+1}$ , debemos mirar ~~que~~ cuál fue la instrucción  $i_j$  que se ejecutó.

Pero sabemos que las instrucciones del lenguaje  $S$  no pueden incrementar una variable en mas de una unidad. Por lo tanto  ~~$s_{j+1}[Y] \leq s_j[Y] + 1$~~

$$s_{j+1}[Y] \leq s_j[Y] + 1$$

Por lo tanto, en el estado final  $K$   ~~$s_K[Y] \leq s_1[Y] + k$~~  vemos que  $s_K[Y] \leq k \leq t$ . Por lo que el resultado de la ejecución,  $\phi_x(y) \leq t$ . □

Inducción en  $t$

b.

Sea un par  ~~$\langle x, y \rangle$~~  tal que  $\langle x, y \rangle \in S$ .

Como  $\langle x, y \rangle \in S$ ,  $\phi_x(y) \downarrow$  y  $\text{STP}(y, x, \phi_x(y) - 1)$

~~Por el lema demostrado en el punto a.~~

$$\phi_x(y) \leq \phi_x(y) - 1$$

Absurdo.

Por lo tanto,  $\nexists \langle x, y \rangle$  tq  $\langle x, y \rangle \in S \Rightarrow S = \emptyset$

$\Rightarrow$   $S$  es computable (ce y co-ce)

\* el característico de  $S$  sería  $S(x) = 0 \forall x$ , que es computable. ✓

4.

Sea  $C$  la clase de funciones  $T_q$

$$C = \{ f: \mathbb{N} \rightarrow \mathbb{N} \mid f(y) \neq 1 \forall y \}$$

Vemos que  $T = \{ x \mid \phi_x \in C \}$ , por lo que  $T$  es un conjunto de índices.

Además, vemos que  $T \neq \mathbb{N}$  ya ~~que un programa que~~ que un programa que compute la función constante 1 ~~no~~ está en  $T$ .   
 // ¿por qué es computable?   
 es, que no está en  $T$  es el número de programa que compute 1   
 Y  $T$  no es vacío pues el número del siguiente programa que se cuelga  $\neq 1$  para cualquier entrada no está en  $T$ .

P: [A] IF  $y=0$  GOTO A

Como valen las hipótesis del teorema de Rice,  $T$  no es computable.

Vemos que  $T$  es co-c.e. Para ello debemos demostrar que la siguiente función es computable:  $\exists x \text{ s.t. } \phi_x(y) \neq 1$

$$f(x) = \begin{cases} 1 & \text{si } \phi_x(y) \neq 1 \forall y \\ 1 & \text{sino} \end{cases}$$

~~Basta~~ Basta con mostrar el siguiente programa que la compute:

P: //  $z$  es una tupla  $\langle y, t \rangle$

$y \leftarrow 1$

[A]  $z \leftarrow z + 1$

IF STP( $l(z), X_1, p(z)$ ) = 0 GOTO A

TE FALTA PROBAR

STP(0,  $X_1$ , 0)

Este programa recorre todas las combinaciones de  $\langle \text{entrada}, \text{pasos} \rangle$  hasta encontrar una entrada para la que termine o colgarse si no existe, que es lo que queríamos ~~computar~~ computar. Por lo tanto,  $T$  es co-c.e.



Finalmente, como  $T$  es co-ce pero no computable,  $T$  no es ce.

